

# **Mathematics of Big Data, I**

## **Lecture 7: Kernel PCA, One Class SVMs, and Learning Theory**

**Weiqing Gu**  
Professor of Mathematics  
Director of the Mathematics Clinic

Harvey Mudd College  
Summer 2017

# Today's topics

- **Summary of Expectation-Maximization Algorithm**
- **EM for MAP estimation (Review MAP first)**
- **Kernel PCA**
- **One-Class Support Vector Machines**
- **Lagrange Duality (only if time permits)**

# Today's topics

- **Summary of Expectation-Maximization Algorithm**
- EM for MAP estimation (Review MAP first)
- Kernel PCA
- One-Class Support Vector Machines
- Lagrange Duality (only if time permits)

# Summary on Expectation-Maximization (EM) Algorithm

- Work out details with the students on the board.

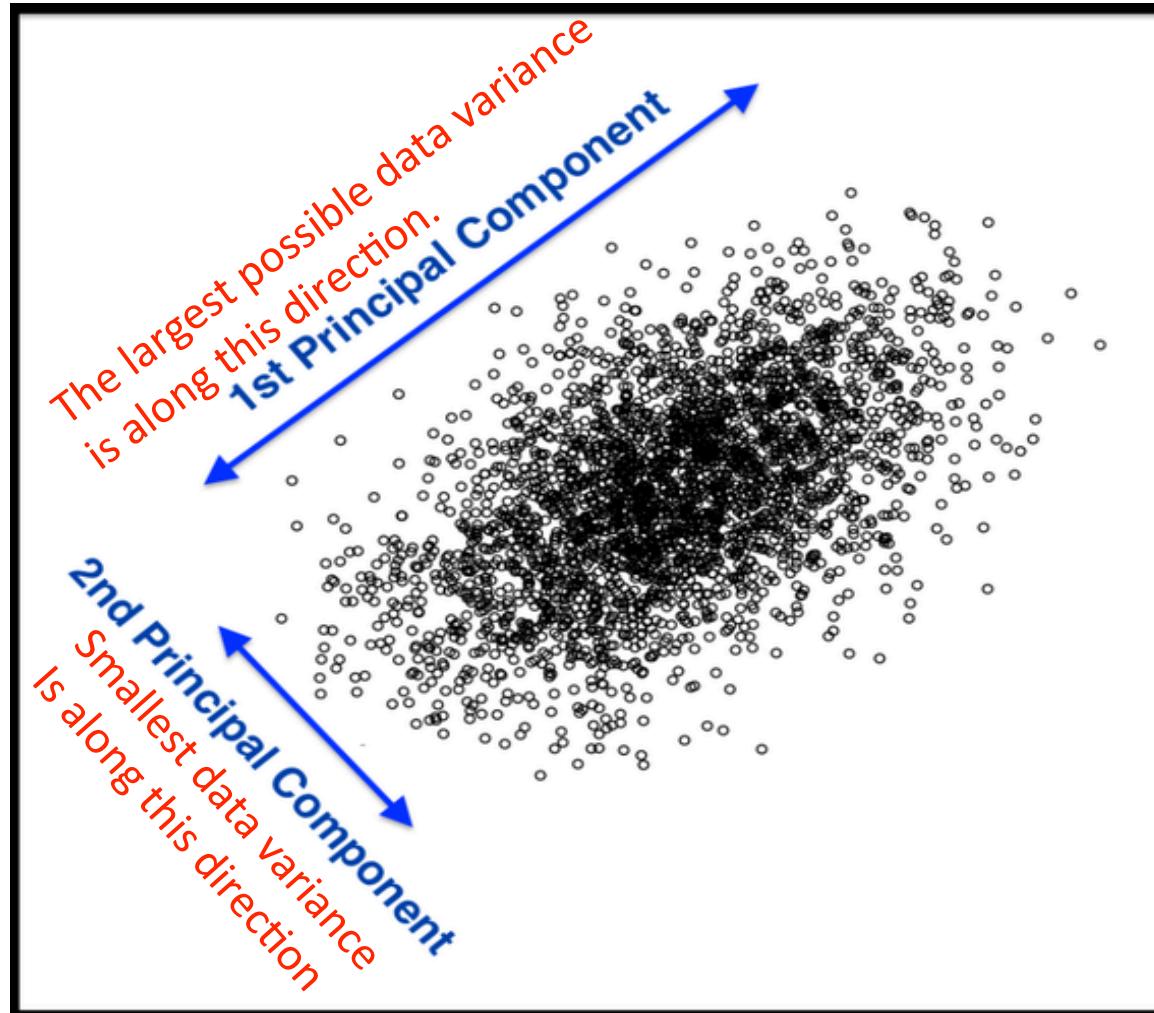
# EM for MAP estimation

- Work out details with the students on the board.
- First review MAP (Maximum A Posteriori)

# Today's topics

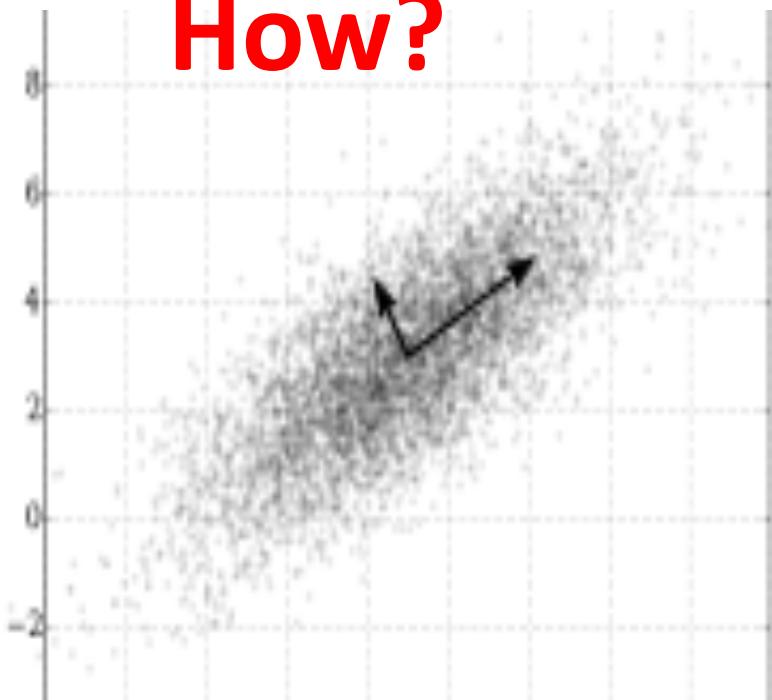
- Summary of Expectation-Maximization Algorithm
- EM for MAP estimation (Review MAP first)
- **Kernel PCA**
- One-Class Support Vector Machines
- Lagrange Duality (only if time permits)

# Recall: PCA



Recall: we want to find two axis directions of the elliptic curve. They are called **principal axes**.

How?



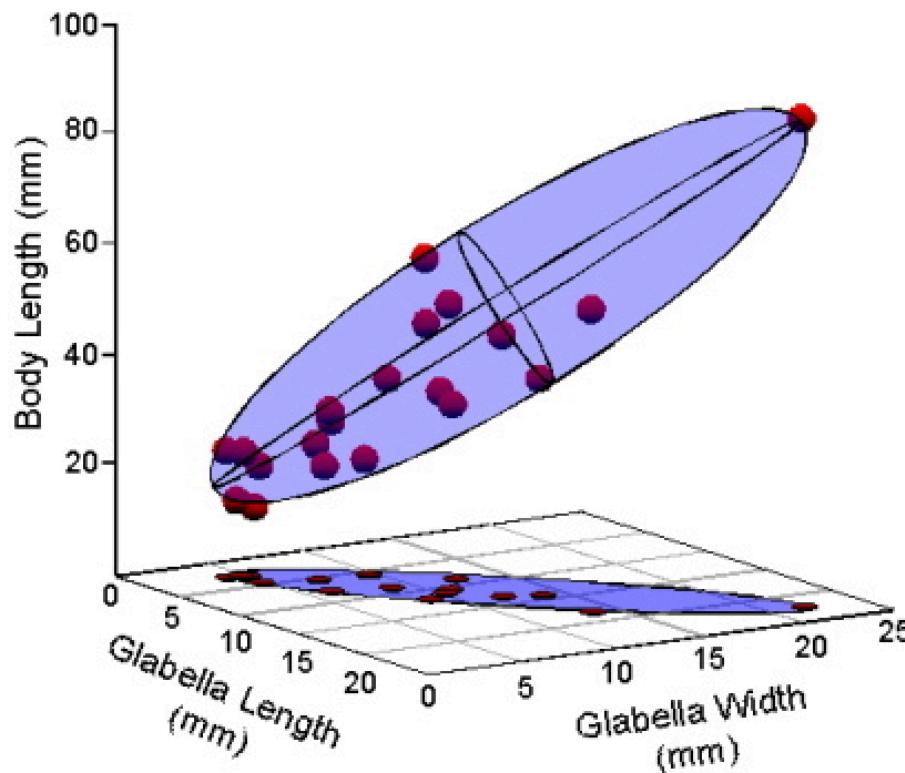
**Note:** the x-value and y-value of the data are correlated.

Their correlation are reflected by the covariance matrix of the data.

- **Principal component analysis (PCA)** is a procedure to find an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated.
- **Procedure of find the principal directions:**
- **Step 1: Find the covariance matrix** of the data directly (Note: one can first standardize data:
  - Find the mean  $\mu_1$  of the x-value & the mean  $\mu_2$  of the y-value.
  - Subtract all x-value by  $\mu_1$  and subtract all y-value by  $\mu_2$ . *Geometrically, move the x-axis and y-axis to the data center.*)
- **Step 2: Find eigenvalues and eigenvectors** of the covariance matrix of the data.
- **Step 3: Order the eigenvalues from largest to smallest.** The eigenvector corresponding to the largest eigenvalue is called the 1<sup>st</sup> principal axis. So on and so forth.
- **Step 4: Form the rotation matrix** using the corresponding eigenvectors.

# Recall: Dimension Reduction Use PCA

- Find the Principal axis and then project to a lower dimensional space.



# Recall: Kernel

$$K = k(\mathbf{x}, \mathbf{y})$$

$$= \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$$

$$= \Phi(\mathbf{x})^T \Phi(\mathbf{y})$$

Example:

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2$$

$$= (\langle \mathbf{x}, \mathbf{y} \rangle + 1)^2$$

Did not calculate directly of  $\Phi(\mathbf{x}), \Phi(\mathbf{y})$

**Gaussian kernel:**

(Often being used)

$$k(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}$$

# Recall: Example of Kernel Method

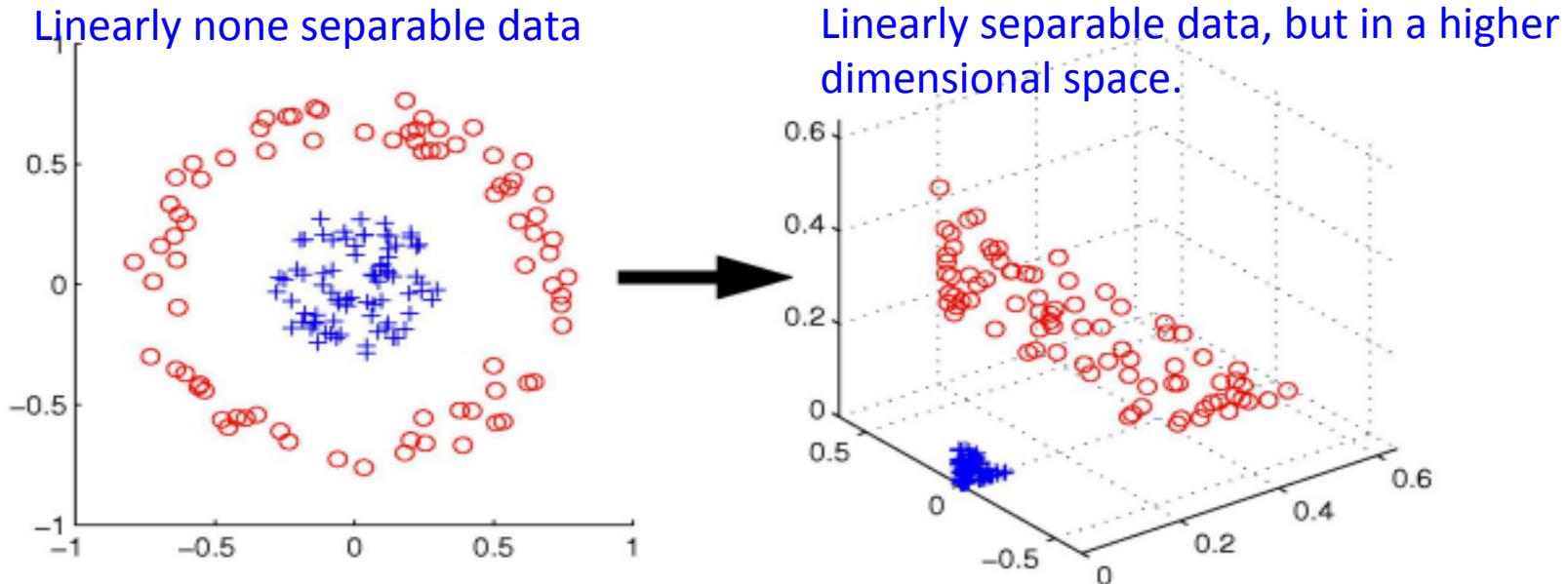


Figure 1: Transforming the data can make it linearly separable

$$\begin{aligned}\phi : \quad \mathbb{R}^2 &\longrightarrow \mathbb{R}^3 \\ (x_1, x_2) &\longmapsto (z_1, z_2, z_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)\end{aligned}$$

$$\begin{aligned}\langle r, s \rangle &= r_1s_1 + r_2s_2 + r_3s_3 \\ &= a_1^2b_1^2 + 2a_1a_2b_1b_2 + a_2^2b_2^2 \\ &= \langle a, b \rangle^2\end{aligned}$$

A function of the original inner product in  $\mathbb{R}^2$ .

# What is kernel PCA?

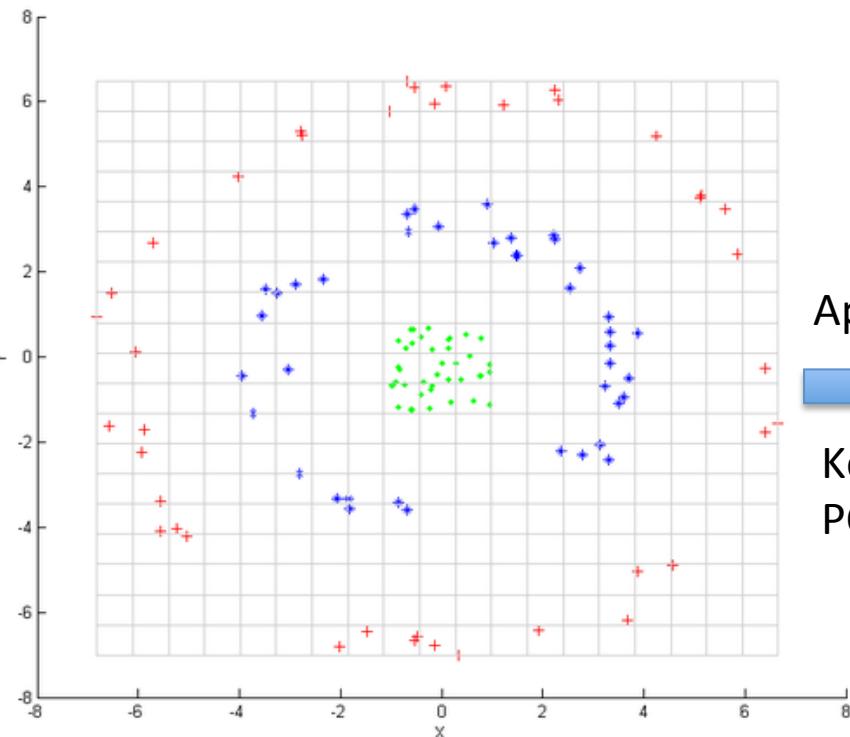
- Kernel PCA =kernel principal component analysis  
= an extension of principal component analysis using techniques of kernel methods.

**Key:** The kernel-formulation of PCA is restricted in that it computes not the principal components themselves, but the projections of our data onto those components.

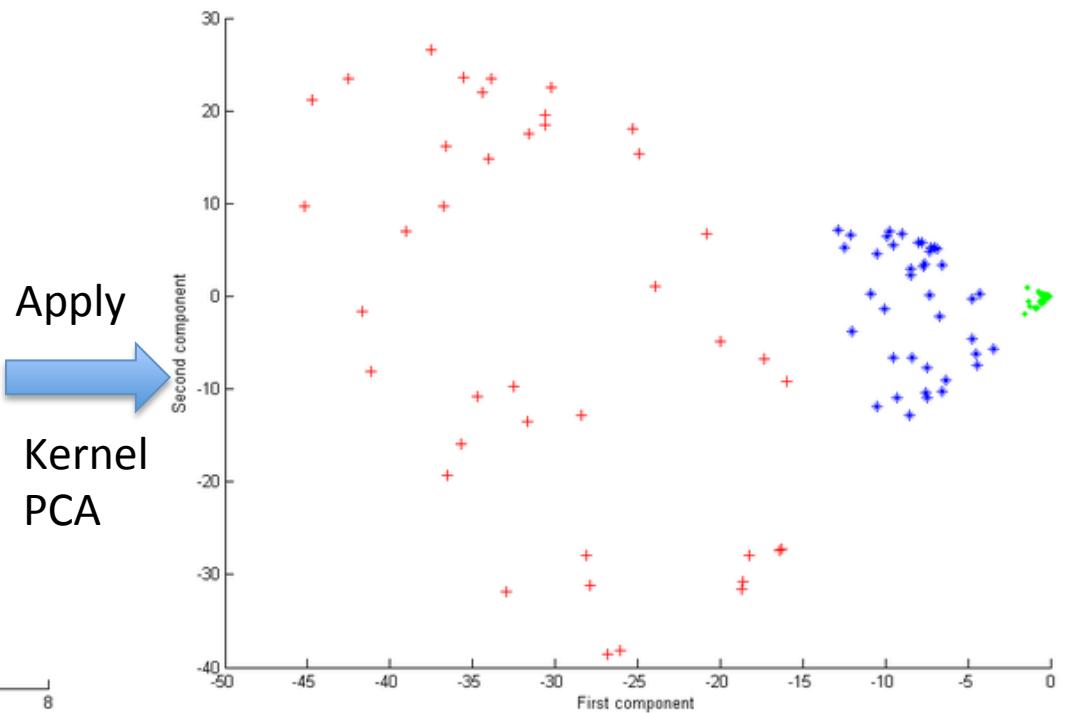
# Example of Kernel PCA: Consider Kernel

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2$$

Linearly non separable data



Separable using the first component



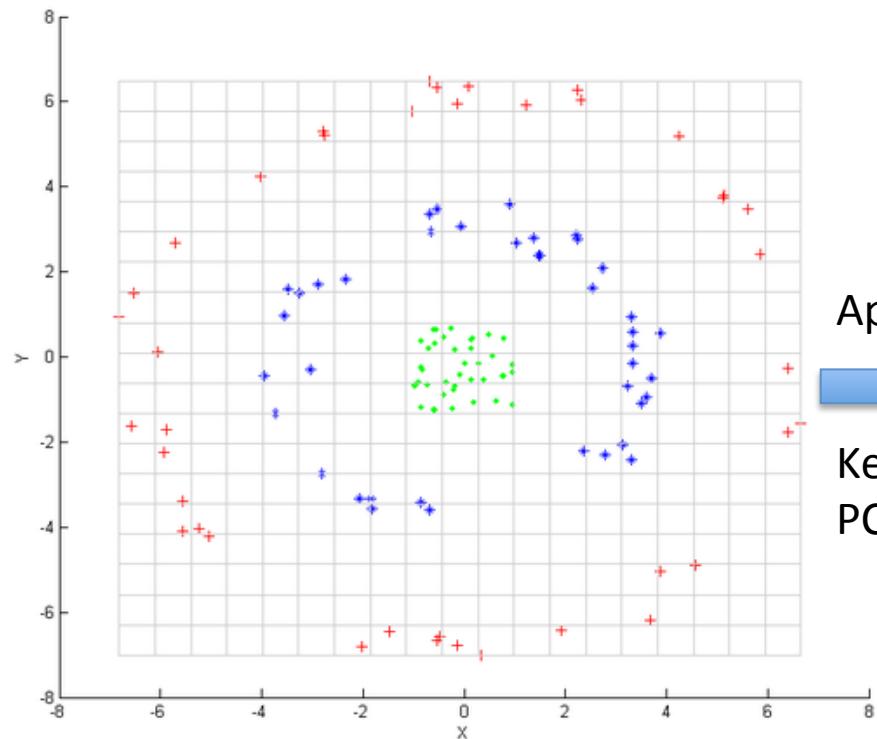
Input points before kernel PCA

Output after kernel PCA in the space of the first and second components.

# Example: Consider Gaussian Kernel

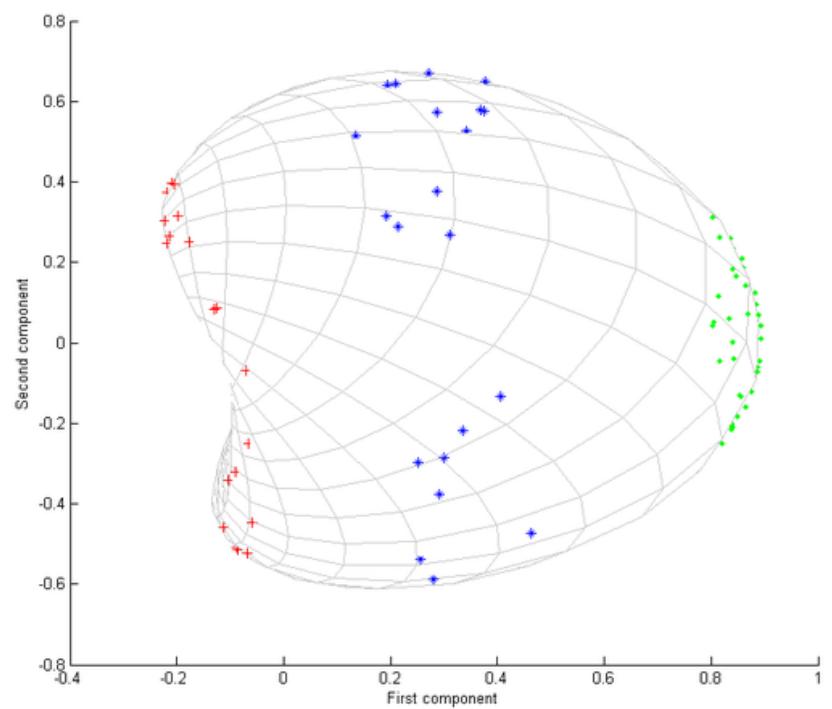
$$k(\mathbf{x}, \mathbf{y}) = e^{\frac{-||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}}$$

Linearly none separable data



Input points before kernel PCA

Separable using the first component



Apply  
Kernel  
PCA

Output after kernel PCA in the space of the first and second components.

# Key idea of Kernel PCA

Cleverly avoid working directly in feature space

Key: the kernel-formulation of PCA is restricted in that it computes not the principal components themselves, but the projections of our data onto those components.

*we never actually solve the eigenvectors and eigenvalues of the covariance matrix in the high dimensional feature space.*

Work out details with students on the board.

## Recall: Gram Matrix

$$\begin{bmatrix} x_1^T x_1 & x_1^T x_2 & \cdots \\ x_2^T x_1 & \ddots \\ \vdots & & \end{bmatrix}_{n*n} = X X^T$$

where  $X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}_{n*d}$

$X$ , containing all the data, is the design matrix

# Kernel Matrix

$$K = \begin{bmatrix} \phi(x_1)^T \phi(x_1) & \phi(x_1)^T \phi(x_2) & \cdots \\ \phi(x_2)^T \phi(x_1) & \ddots & \\ \vdots & & \end{bmatrix}$$

Note: we first map our data via some function  $\phi$ , then form the Gram Matrix.

# What is Kernel Trick?

Cleverly avoid working directly in high dim'l feature space.

$$K = k(\mathbf{x}, \mathbf{y})$$

$$= \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$$

$$= \Phi(\mathbf{x})^T \Phi(\mathbf{y})$$

Example:

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2$$

$$= (\langle \mathbf{x}, \mathbf{y} \rangle + 1)^2$$



Did not calculate directly of  $\Phi(\mathbf{x}), \Phi(\mathbf{y})$

But use the kernel function  $k(\mathbf{x}, \mathbf{y})$   
to only calculate  $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$

- Kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space.

- This operation is often computationally cheaper than the explicit computation of the coordinates. This approach is called the "kernel trick". Kernel functions have been introduced for sequence data, graphs, text, images, as well as vectors

Key: We do not need to write down exactly what is  $\Phi$ , but the result of  $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$

It suffices for us to know such a  $\Phi$  exit.

We check it by using **Mercer's Theorem**

A symmetric function  $K(x, y)$  can be expressed as an inner product

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

for some  $\phi$  if and only if  $K(x, y)$  is positive semidefinite, i.e.

$$\int K(x, y)g(x)g(y)dxdy \geq 0 \quad \forall g$$

or, equivalently:

$$\begin{bmatrix} K(x_1, x_1) & K(x_1, x_2) & \cdots \\ K(x_2, x_1) & \ddots & \\ \vdots & & \end{bmatrix} \text{ is psd for any collection } \{x_1 \dots x_n\}$$

# Applications of Kernel Methods

- We can turn a linear model into a non-linear model by applying the kernel trick to the model: replacing its features (predictors) by a kernel function.
- The following algorithms are capable of operating with the kernel method:
  - Support vector machines (SVM)
  - Gaussian processes
  - Principal components analysis (PCA)
  - Canonical correlation analysis
  - Ridge regression
  - Spectral clustering
  - Linear adaptive filters
  - The kernel perceptron
  - And many others.

# Today's topics

- Summary of Expectation-Maximization Algorithm
- EM for MAP estimation (Review MAP first)
- Kernel PCA
- **One-Class Support Vector Machines**
- Lagrange Duality (only if time permits)

# What is One-class Support Vector Machines?

- Recall: The goal of the machine learning application is to distinguish test data between a number of classes, using training data.
- But what if you only **have data of one class** and the goal is to test new data and found out whether it is alike or not like the training data?
- A method for this task is the **One-Class Support Vector Machine**.

# Today's topics

- Summary of Expectation-Maximization Algorithm
- EM for MAP estimation (Review MAP first)
- Kernel PCA
- One-Class Support Vector Machines
- **Lagrange Duality (only if time permits)**

# Lagrange Duality

(Please read the rest of the slides)

Consider a general problem:

$$\begin{aligned} \min_w \quad & f(w) \\ \text{s.t.} \quad & h_i(w) = 0, \quad i = 1, \dots, l. \end{aligned}$$

Define **the Lagrangian**:

$$\mathcal{L}(w, \beta) = f(w) + \sum_{i=1}^l \beta_i h_i(w)$$

Here, the  $\beta_i$ 's are called the *Lagrange multipliers*.

# Key steps involved

- Find the partial derivatives and set them to zero.

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0; \quad \frac{\partial \mathcal{L}}{\partial \beta_i} = 0,$$

- Solve for  $w$  and  $\beta$ .
- *Q: How to deal with constrained optimization problems in which there are inequality as well as equality constraint?*

Consider

$$\min_w \quad f(w)$$

$$\text{s.t.} \quad g_i(w) \leq 0, \quad i = 1, \dots, k$$

$$h_i(w) = 0, \quad i = 1, \dots, l.$$

This optimization problem is called  
**the primal optimization problem.**

# Generalized Lagrangian

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w).$$

Here, the  $\alpha_i$ 's and  $\beta_i$ 's are the Lagrange multipliers.

Similar as before.

# Key: How to get rid of the inequalities?

Consider:

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta : \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta).$$



Primal optimization

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta : \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta).$$

Here, the “ $\mathcal{P}$ ” subscript stands for “primal.” Let some  $w$  be given. If  $w$  violates any of the primal constraints (i.e., if either  $g_i(w) > 0$  or  $h_i(w) \neq 0$  for some  $i$ ), then you should be able to verify that

$$\begin{aligned}\theta_{\mathcal{P}}(w) &= \max_{\alpha, \beta : \alpha_i \geq 0} f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w) \quad (1) \\ &= \infty.\end{aligned}$$

Conversely, if the constraints are indeed satisfied for a particular value of  $w$ , then  $\theta_{\mathcal{P}}(w) = f(w)$ . Hence,

$$\theta_{\mathcal{P}}(w) = \begin{cases} f(w) & \text{if } w \text{ satisfies primal constraints} \\ \infty & \text{otherwise.} \end{cases}$$

# We essentially be able to get rid of the inequalities.

Thus,  $\theta_{\mathcal{P}}$  takes the same value as the objective in our problem for all values of  $w$  that satisfies the primal constraints, and is positive infinity if the constraints are violated. Hence, if we consider the minimization problem

$$\min_w \theta_{\mathcal{P}}(w) = \min_w \max_{\alpha, \beta : \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta),$$

we see that it is the same problem (i.e., and has the same solutions as) our original, primal problem. For later use, we also define the optimal value of the objective to be  $p^* = \min_w \theta_{\mathcal{P}}(w)$ ; we call this the **value** of the primal problem.

# The Dual Optimization Problem

Now, let's look at a slightly different problem. We define

$$\theta_{\mathcal{D}}(\alpha, \beta) = \min_w \mathcal{L}(w, \alpha, \beta).$$

Here, the “ $\mathcal{D}$ ” subscript stands for “dual.” Note also that whereas in the definition of  $\theta_{\mathcal{P}}$  we were optimizing (maximizing) with respect to  $\alpha, \beta$ , here we are minimizing with respect to  $w$ .

We can now pose the **dual** optimization problem:

$$\max_{\alpha, \beta : \alpha_i \geq 0} \theta_{\mathcal{D}}(\alpha, \beta) = \max_{\alpha, \beta : \alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta).$$

This is exactly the same as our primal problem shown above, except that the order of the “max” and the “min” are now exchanged. We also define the optimal value of the dual problem’s objective to be  $d^* = \max_{\alpha, \beta : \alpha_i \geq 0} \theta_{\mathcal{D}}(w)$ .

# Relations between Max-Min and Min-Max Problems

How are the primal and the dual problems related? It can easily be shown that

$$d^* = \max_{\alpha, \beta: \alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta) \leq \min_w \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta) = p^*.$$

(You should convince yourself of this; this follows from the “max min” of a function always being less than or equal to the “min max.”) However, under certain conditions, we will have

*The proof is out of scope of  
This course.*

$$d^* = p^*,$$

so that we can solve the dual problem in lieu of the primal problem. Let's see what these conditions are.

# Q: When do the primal and dual problems have same solution?

Suppose  $f$  and the  $g_i$ 's are convex,<sup>3</sup> and the  $h_i$ 's are affine.<sup>4</sup> Suppose further that the constraints  $g_i$  are (strictly) feasible; this means that there exists some  $w$  so that  $g_i(w) < 0$  for all  $i$ .

Under our above assumptions, there must exist  $w^*, \alpha^*, \beta^*$  so that  $w^*$  is the solution to the primal problem,  $\alpha^*, \beta^*$  are the solution to the dual problem, and moreover  $p^* = d^* = \mathcal{L}(w^*, \alpha^*, \beta^*)$ . Moreover,  $w^*, \alpha^*$  and  $\beta^*$  satisfy the **Karush-Kuhn-Tucker (KKT) conditions**

<sup>3</sup>When  $f$  has a Hessian, then it is convex if and only if the Hessian is positive semi-definite. For instance,  $f(w) = w^T w$  is convex; similarly, all linear (and affine) functions are also convex. (A function  $f$  can also be convex without being differentiable, but we won't need those more general definitions of convexity here.)

<sup>4</sup>I.e., there exists  $a_i, b_i$ , so that  $h_i(w) = a_i^T w + b_i$ . “Affine” means the same thing as linear, except that we also allow the extra intercept term  $b_i$ .

# KKT Conditions

Karush-Kuhn-Tucker (KKT) conditions, which are as follows:

$$\frac{\partial}{\partial w_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \dots, n \quad (3)$$

$$\frac{\partial}{\partial \beta_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \dots, l \quad (4)$$

$$\alpha_i^* g_i(w^*) = 0, \quad i = 1, \dots, k \quad (5)$$

$$g_i(w^*) \leq 0, \quad i = 1, \dots, k \quad (6)$$

$$\alpha^* \geq 0, \quad i = 1, \dots, k \quad (7)$$

Moreover, if some  $w^*, \alpha^*, \beta^*$  satisfy the KKT conditions, then it is also a solution to the primal and dual problems.

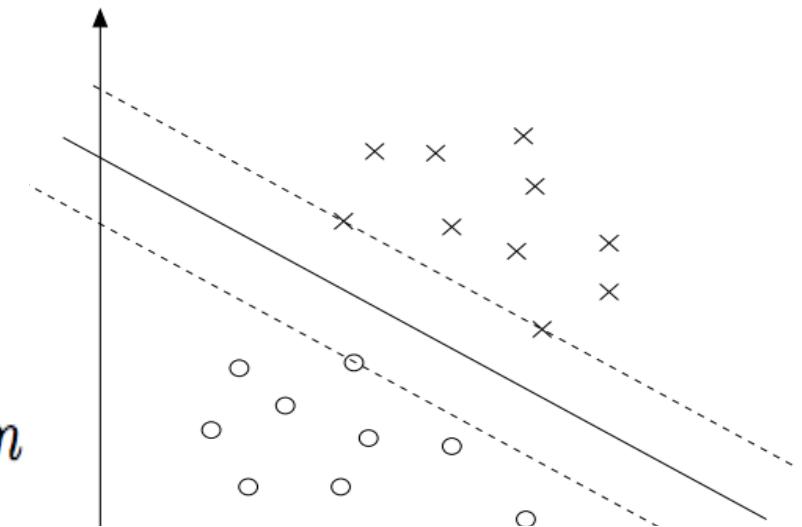
# KKT dual complementarity condition

$$\alpha_i^* g_i(w^*) = 0, \quad i = 1, \dots, k \quad (5)$$

We draw attention to Equation (5), which is called the **KKT dual complementarity** condition. Specifically, it implies that if  $\alpha_i^* > 0$ , then  $g_i(w^*) = 0$ . (I.e., the “ $g_i(w) \leq 0$ ” constraint is **active**, meaning it holds with equality rather than with inequality.) Later on, this will be key for showing that the SVM has only a small number of “support vectors”; the KKT dual complementarity condition will also give us our convergence test when we talk about the SMO algorithm.

# Optimal margin classifiers--SVM

$$\begin{aligned} & \min_{\gamma, w, b} \frac{1}{2} \|w\|^2 \\ \text{s.t. } & y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \dots, m \end{aligned}$$



We can write the constraints as

$$g_i(w) = -y^{(i)}(w^T x^{(i)} + b) + 1 \leq 0.$$

We have one such constraint for each training example. Note that from the KKT dual complementarity condition, we will have  $\alpha_i > 0$  only for the training examples that have functional margin exactly equal to one (i.e., the ones corresponding to constraints that hold with equality,  $g_i(w) = 0$ ). Consider the figure below, in which a maximum margin separating hyperplane is shown by the solid line.

# Back to SVM

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} ||w||^2 - \sum_{i=1}^m \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1].$$

Note that there're only “ $\alpha_i$ ” but no “ $\beta_i$ ” Lagrange multipliers, since the problem has only inequality constraints.

Let's find the dual form of the problem. To do so, we need to first minimize  $\mathcal{L}(w, b, \alpha)$  with respect to  $w$  and  $b$  (for fixed  $\alpha$ ), to get  $\theta_D$ , which we'll do by setting the derivatives of  $\mathcal{L}$  with respect to  $w$  and  $b$  to zero. We have:

$$\nabla_w \mathcal{L}(w, b, \alpha) = w - \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} = 0$$

This implies that

$$w = \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}. \quad (9)$$

As for the derivative with respect to  $b$ , we obtain

$$\frac{\partial}{\partial b} \mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i y^{(i)} = 0. \quad (10)$$

If we take the definition of  $w$  in Equation (9) and plug that back into the Lagrangian (Equation 8), and simplify, we get

$$\mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T x^{(j)} - b \sum_{i=1}^m \alpha_i y^{(i)}.$$

But from Equation (10), the last term must be zero, so we obtain

$$\mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T x^{(j)}.$$

# Dual Optimization Problem-SVM

Recall that we got to the equation above by minimizing  $\mathcal{L}$  with respect to  $w$  and  $b$ . Putting this together with the constraints  $\alpha_i \geq 0$  (that we always had) and the constraint (10), we obtain the following dual optimization problem:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle.$$

$$\text{s.t. } \alpha_i \geq 0, \quad i = 1, \dots, m$$

$$\sum_{i=1}^m \alpha_i y^{(i)} = 0,$$



Key: All the x-data is in this inner product!

# From this dual point view, we will see SVM can be generated by replacing the inner product by a Kernel!

By examining the dual form of the optimization problem, we gained significant insight into the structure of the problem, and were also able to write the entire algorithm in terms of only inner products between input feature vectors. In the next section, we will exploit this property to apply the kernels to our classification problem. The resulting algorithm, **support vector machines**, will be able to efficiently learn in very high dimensional spaces.

# Exercise

You should also be able to verify that the conditions required for  $p^* = d^*$  and the KKT conditions (Equations 3–7) to hold are indeed satisfied in our optimization problem. Hence, we can solve the dual in lieu of solving the primal problem. Specifically, in the dual problem above, we have a maximization problem in which the parameters are the  $\alpha_i$ 's. We'll talk later about the specific algorithm that we're going to use to solve the dual problem, but if we are indeed able to solve it (i.e., find the  $\alpha$ 's that maximize  $W(\alpha)$  subject to the constraints), then we can use Equation (9) to go back and find the optimal  $w$ 's as a function of the  $\alpha$ 's. Having found  $w^*$ , by considering the primal problem, it is also straightforward to find the optimal value for the intercept term  $b$  as

$$b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + \min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2}. \quad (11)$$

# Cool thing about this dual method

Before moving on, let's also take a more careful look at Equation (9), which gives the optimal value of  $w$  in terms of (the optimal value of)  $\alpha$ . Suppose we've fit our model's parameters to a training set, and now wish to make a prediction at a new point input  $x$ . We would then calculate  $w^T x + b$ , and predict  $y = 1$  if and only if this quantity is bigger than zero. But using (9), this quantity can also be written:

$$w^T x + b = \left( \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} \right)^T x + b \quad (12)$$

$$= \sum_{i=1}^m \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b. \quad (13)$$

Hence, if we've found the  $\alpha_i$ 's, in order to make a prediction, we have to calculate a quantity that depends only on the inner product between  $x$  and the points in the training set. Moreover, we saw earlier that the  $\alpha_i$ 's will all be zero except for the support vectors. Thus, many of the terms in the sum above will be zero, and we really need to find only the inner products between  $x$  and the support vectors (of which there is often only a small number) in order calculate (13) and make our prediction.

- Back up slides

# Kernel Trick

Cleverly avoid working directly in feature space

$$K = k(\mathbf{x}, \mathbf{y})$$

$$= \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$$

$$= \Phi(\mathbf{x})^T \Phi(\mathbf{y})$$

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2$$

$$= (\langle \mathbf{x}, \mathbf{y} \rangle + 1)^2$$



- Kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space.

This operation is often computationally cheaper than the explicit computation of the coordinates. This approach is called the "kernel trick". Kernel functions have been introduced for sequence data, graphs, text, images, as well as vectors.

# correlation coefficient & correlation matrix

- The (Pearson) **correlation coefficient** between two rvs  $X$  and  $Y$  is defined as

$$\text{corr} [X, Y] \triangleq \frac{\text{cov} [X, Y]}{\sqrt{\text{var} [X] \text{var} [Y]}}$$

- If  $X$  and  $Y$  are indep., then  $\text{cov} [X, Y] = 0$ ; say  $X$  and  $Y$  are uncorrelated.
- A **correlation matrix** of a random vector has the form:

$$\mathbf{R} = \begin{pmatrix} \text{corr} [X_1, X_1] & \text{corr} [X_1, X_2] & \cdots & \text{corr} [X_1, X_d] \\ \vdots & \vdots & \ddots & \vdots \\ \text{corr} [X_d, X_1] & \text{corr} [X_d, X_2] & \cdots & \text{corr} [X_d, X_d] \end{pmatrix}$$

Exercise: show that  $-1 \leq \text{corr} [X, Y] \leq 1$  and  
Show that  $\text{corr}[X,Y] = 1$  iff  $Y = aX + b$  for some parameters  $a$  and  $b$ .